



10. LINEAR PROGRAMMING

Raveen de Silva, r.desilva@unsw.edu.au

office: K17 202

Course Admin: Anahita Namvar, cs3121@cse.unsw.edu.au

School of Computer Science and Engineering
UNSW Sydney

Term 2, 2022

Table of Contents

1. Example Problems
2. Linear Programming
3. Puzzle

Linear Programming problems: Example 1

Problem

Instance: a list of food sources F_1, \dots, F_n ; and for each source F_i :

- its price per gram p_i ;
- the number of calories c_i per gram, and
- for each of 13 vitamins V_1, \dots, V_{13} , the content $v_{i,j}$ in milligrams of vitamin V_j in one gram of food source f_i .

Task: find a combination of quantities of food sources such that:

- the total number of calories in all of the chosen food is equal to a recommended daily value of 2000 calories;
- for each $1 \leq j \leq 13$, the total intake of vitamin V_j is at least the recommended daily intake of w_j milligrams, and
- the price of all food per day is as low as possible.

Linear Programming problems: Example 1

Suppose we take x_i grams of each food source F_i for $1 \leq i \leq n$. Then the constraints are as follows.

- The total number of calories must satisfy

$$\sum_{i=1}^n x_i c_i = 2000;$$

- For each $1 \leq j \leq 13$, the total amount of vitamin V_j in all food must satisfy

$$\sum_{i=1}^n x_i v_{i,j} \geq w_j.$$

- Implicitly, all the quantities must be non-negative numbers, i.e. $x_i \geq 0$ for all $1 \leq i \leq n$.

Linear Programming problems: Example 1

- Our goal is to minimise the objective function, which is the total cost

$$y = \sum_{i=1}^n x_i p_i.$$

- Note that all constraints and the objective function are **linear**.

Linear Programming problems: Example 2

Problem

Instance: you are a politician and you want to ensure an election victory by making certain promises to the electorate. You can promise to build:

- bridges, each costing 3 billion;
- rural airports, each costing 2 billion, and
- Olympic swimming pools, each costing 1 billion.

Linear Programming problems: Example 2

Problem (continued)

You were told by your wise advisers that

- each bridge you promise brings you 5% of city votes, 7% of suburban votes and 9% of rural votes;
- each rural airport you promise brings you no city votes, 2% of suburban votes and 15% of rural votes;
- each Olympic swimming pool promised brings you 12% of city votes, 3% of suburban votes and no rural votes.

Linear Programming problems: Example 2

Problem (continued)

In order to win, you have to get at least 51% of each of the city, suburban and rural votes.

Task: decide how many bridges, airports and pools to promise in order to guarantee an election win at minimum cost to the budget.

Linear Programming problems: Example 2

- Let the number of bridges to be built be x_b , number of airports x_a and the number of swimming pools x_p .
- We now see that the problem amounts to minimising the objective $y = 3x_b + 2x_a + x_p$, while making sure that the following constraints are satisfied:

$$0.05x_b + 0.12x_p \geq 0.51 \quad (\text{city votes})$$

$$0.07x_b + 0.02x_a + 0.03x_p \geq 0.51 \quad (\text{suburban votes})$$

$$0.09x_b + 0.15x_a \geq 0.51 \quad (\text{rural votes})$$

$$x_b, x_a, x_p \geq 0.$$

Linear Programming problems: Example 2

- However, there is a very significant difference with the first example:
 - you can eat 1.56 grams of chocolate, but
 - you cannot promise to build 1.56 bridges, 2.83 airports and 0.57 swimming pools!
- The second example is an example of an **Integer Linear Programming problem**, which requires all the solutions to be integers.
- Such problems are MUCH harder to solve than the “plain” Linear Programming problems whose solutions can be real numbers.

Solving Linear Programming problems

- We won't see algorithms which solve LP problems in this lecture; we will only study the structure of these problems further.
- There are polynomial time algorithms for Linear Programming, including the ellipsoid algorithm.
- In practice we typically use the SIMPLEX algorithm instead; its worst case time complexity is exponential, but it is very efficient in the 'average' case.
- There is no known polynomial time algorithm for Integer Linear Programming!

Table of Contents

1. Example Problems
2. Linear Programming
3. Puzzle

Standard Form

In the **standard form** the *objective* to be maximised is given by

$$\sum_{j=1}^n c_j x_j$$

and the *constraints* are of the form

$$\begin{aligned} \sum_{j=1}^n a_{ij} x_j &\leq b_i & (1 \leq i \leq m); \\ x_j &\geq 0 & (1 \leq j \leq n). \end{aligned}$$

Standard Form

- To get a more compact representation of linear programs, we use vectors and matrices.
- Let \mathbf{x} represent a (column) vector,

$$\mathbf{x} = \langle x_1 \dots x_n \rangle^T.$$

- Define a partial ordering on the vectors in \mathbb{R}^n by $\mathbf{x} \leq \mathbf{y}$ if and only if the corresponding inequalities hold coordinate-wise, i.e., if and only if $x_j \leq y_j$ for all $1 \leq j \leq n$.

Standard Form

Write the coefficients in the objective function as

$$\mathbf{c} = \langle c_1 \dots c_n \rangle^T \in \mathbb{R}^n,$$

the coefficients in the constraints as an $m \times n$ matrix

$$A = (a_{ij})$$

and the right-hand side values of the constraints as

$$\mathbf{b} = \langle b_1 \dots b_m \rangle^T \in \mathbb{R}^m.$$

Standard Form

Then the standard form can be formulated simply as:

- maximize $\mathbf{c}^T \mathbf{x}$
- subject to the following two (matrix-vector) constraints:

$$A\mathbf{x} \leq \mathbf{b}$$

$$\mathbf{x} \geq \mathbf{0}.$$

Thus, a Linear Programming optimisation problem can be specified as a triplet $(A, \mathbf{b}, \mathbf{c})$, which is the form accepted by most standard LP solvers.

Translating other constraints to Standard Form

- The Standard Form doesn't immediately appear to handle the full generality of LP problems.
- LP problems could have:
 - equality constraints
 - unconstrained variables (i.e. potentially negative values x_i)
 - absolute value constraints

Equality constraints

- An LP problem may include equality constraints of the form

$$\sum_{i=1}^n a_{ij}x_i = b_j.$$

- Each of can be replaced by two inequalities:

$$\sum_{i=1}^n a_{ij}x_i \geq b_j$$

$$\sum_{i=1}^n a_{ij}x_i \leq b_j.$$

- Thus, we can assume that all constraints are inequalities.

Unconstrained variables

- In general, a “natural formulation” of a problem as a Linear Program does not necessarily require that all variables be non-negative.
- However, the Standard Form does impose this constraint.
- This poses no problem, because each occurrence of an unconstrained variable x_j can be replaced by the expression

$$x'_j - x_j^*$$

where x'_j, x_j^* are new variables satisfying the inequality constraints

$$x'_j \geq 0, \quad x_j^* \geq 0.$$

Absolute value constraints

- For a vector

$$\mathbf{x} = \langle x_1, \dots, x_n \rangle^T,$$

we can define

$$|\mathbf{x}| = \langle |x_1|, \dots, |x_n| \rangle^T.$$

- Some problems are naturally translated into constraints of the form

$$|\mathbf{Ax}| \leq \mathbf{b}.$$

- This also poses no problem because we can replace such constraints with two linear constraints:

$$\mathbf{Ax} \leq \mathbf{b} \text{ and } -\mathbf{Ax} \leq \mathbf{b},$$

because $|x| \leq y$ if and only if $x \leq y$ and $-x \leq y$.

Summary of Standard Form

- Standard Form: maximize

$$\mathbf{c}^T \mathbf{x}$$

subject to

$$A\mathbf{x} \leq \mathbf{b}$$

and

$$\mathbf{x} \geq \mathbf{0}.$$

- Any vector \mathbf{x} which satisfies the two constraints is called a *feasible solution*, regardless of what the corresponding objective value $\mathbf{c}^T \mathbf{x}$ might be.

Sample linear program

As an example, let us consider the following optimisation problem.

Problem

$$\text{maximise} \quad z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3 \quad (1)$$

subject to

$$x_1 + x_2 + 3x_3 \leq 30 \quad (2)$$

$$2x_1 + 2x_2 + 5x_3 \leq 24 \quad (3)$$

$$4x_1 + x_2 + 2x_3 \leq 36 \quad (4)$$

$$x_1, x_2, x_3 \geq 0 \quad (5)$$

Sample linear program

How large can the value of the objective

$$z(x_1, x_2, x_3) = 3x_1 + x_2 + 2x_3$$

be, without violating the constraints?

We can achieve a crude bound by adding inequalities (2) and (3), to obtain

$$3x_1 + 3x_2 + 8x_3 \leq 54.$$

Since all variables are constrained to be non-negative, we are assured that

$$3x_1 + x_2 + 2x_3 \leq 3x_1 + 3x_2 + 8x_3 \leq 54,$$

i.e. the objective does not exceed 54. Can we do better?

Sample linear program

We could try to look for coefficients $y_1, y_2, y_3 \geq 0$ to be used to form a linear combination of the constraints:

$$y_1(x_1 + x_2 + 3x_3) \leq 30y_1 \quad (6)$$

$$y_2(2x_1 + 2x_2 + 5x_3) \leq 24y_2 \quad (7)$$

$$y_3(4x_1 + x_2 + 2x_3) \leq 36y_3 \quad (8)$$

Then, summing up all these inequalities and factoring, we get

$$\begin{aligned} & x_1(y_1 + 2y_2 + 4y_3) \\ & + x_2(y_1 + 2y_2 + y_3) \\ & + x_3(3y_1 + 5y_2 + 2y_3) \\ & \leq 30y_1 + 24y_2 + 36y_3. \end{aligned}$$

Sample linear program

If we compare this with our objective (1) we see that if we choose y_1, y_2 and y_3 so that:

$$y_1 + 2y_2 + 4y_3 \geq 3$$

$$y_1 + 2y_2 + y_3 \geq 1$$

$$3y_1 + 5y_2 + 2y_3 \geq 2$$

then

$$\begin{aligned} 3x_1 + x_2 + 2x_3 &\leq x_1(y_1 + 2y_2 + 4y_3) \\ &\quad + x_2(y_1 + 2y_2 + y_3) \\ &\quad + x_3(3y_1 + 5y_2 + 2y_3). \end{aligned}$$

Combining this with (6) – (8) we get

$$30y_1 + 24y_2 + 36y_3 \geq 3x_1 + x_2 + 2x_3 = z(x_1, x_2, x_3).$$

Sample linear program

Consequently, in order to find a tight upper bound for our objective $z(x_1, x_2, x_3)$ in the original problem P , we have to find y_1, y_2, y_3 which solve problem P^* :

$$\text{minimise:} \quad z^*(y_1, y_2, y_3) = 30y_1 + 24y_2 + 36y_3 \quad (9)$$

subject to:

$$y_1 + 2y_2 + 4y_3 \geq 3 \quad (10)$$

$$y_1 + 2y_2 + y_3 \geq 1 \quad (11)$$

$$3y_1 + 5y_2 + 2y_3 \geq 2 \quad (12)$$

$$y_1, y_2, y_3 \geq 0 \quad (13)$$

Sample linear program

Then

$$\begin{aligned} z^*(y_1, y_2, y_3) &= 30y_1 + 24y_2 + 36y_3 \\ &\geq 3x_1 + x_2 + 2x_3 \\ &= z(x_1, x_2, x_3) \end{aligned}$$

will be a tight upper bound.

The new problem P^* is called the *dual problem* of P .

Sample linear program

Let us now repeat the whole procedure in order to find the dual of P^* , which will be denoted $(P^*)^*$.

We are now looking for $z_1, z_2, z_3 \geq 0$ to multiply inequalities (10)–(12) and obtain

$$\begin{aligned}z_1(y_1 + 2y_2 + 4y_3) &\geq 3z_1 \\z_2(y_1 + 2y_2 + y_3) &\geq z_2 \\z_3(3y_1 + 5y_2 + 2y_3) &\geq 2z_3\end{aligned}$$

Summing these up and factoring produces

$$\begin{aligned}&y_1(z_1 + z_2 + 3z_3) \\&+ y_2(2z_1 + 2z_2 + 5z_3) \\&+ y_3(4z_1 + z_2 + 2z_3) \\&\geq 3z_1 + z_2 + 2z_3\end{aligned}\tag{14}$$

Sample linear program

If we choose multipliers z_1, z_2, z_3 so that

$$z_1 + z_2 + 3z_3 \leq 30$$

$$2z_1 + 2z_2 + 5z_3 \leq 24$$

$$4z_1 + z_2 + 2z_3 \leq 36$$

we will have:

$$\begin{aligned} & y_1(z_1 + z_2 + 3z_3) \\ & + y_2(2z_1 + 2z_2 + 5z_3) \\ & + y_3(4z_1 + z_2 + 2z_3) \\ & \leq 30y_1 + 24y_2 + 36y_3 \end{aligned}$$

Combining this with (14) we get

$$3z_1 + z_2 + 2z_3 \leq 30y_1 + 24y_2 + 36y_3.$$

Sample linear program

Consequently, finding the double dual program $(P^*)^*$ amounts to maximising the objective $3z_1 + z_2 + 2z_3$ subject to the constraints

$$z_1 + z_2 + 3z_3 \leq 30$$

$$2z_1 + 2z_2 + 5z_3 \leq 24$$

$$4z_1 + z_2 + 2z_3 \leq 36$$

$$z_1, z_2, z_3 \geq 0$$

This is exactly our starting program P , with only the variable names changed! Thus, the double dual program $(P^*)^*$ is just P itself.

Sample linear program

- It appeared at first that looking for the multipliers y_1, y_2, y_3 did not help much, because it only reduced a maximisation problem to an equally hard minimisation problem.
- It is useful at this point to remember how we proved that the Ford-Fulkerson algorithm produces a **maximal flow**, by showing that it terminates only when we reach the capacity of a **minimal cut**.

Primal and dual linear programs

In general, the *primal* Linear Program P and its *dual* P^* are:

$$P : \text{maximize} \quad z(\mathbf{x}) = \sum_{j=1}^n c_j x_j,$$

$$\text{subject to} \quad \sum_{j=1}^n a_{ij} x_j \leq b_i \quad (1 \leq i \leq m)$$

$$\text{and} \quad x_1, \dots, x_n \geq 0;$$

$$P^* : \text{minimize} \quad z^*(\mathbf{y}) = \sum_{i=1}^m b_i y_i,$$

$$\text{subject to} \quad \sum_{i=1}^m a_{ij} y_i \geq c_j \quad (1 \leq j \leq n)$$

$$\text{and} \quad y_1, \dots, y_m \geq 0.$$

Primal and dual linear programs

We can equivalently write P and P^* in matrix form:

$$\begin{array}{ll} P : & \text{maximize} \quad z(\mathbf{x}) = \mathbf{c}^T \mathbf{x}, \\ & \text{subject to} \quad A\mathbf{x} \leq \mathbf{b} \\ & \text{and} \quad \mathbf{x} \geq 0; \\ P^* : & \text{minimize} \quad z^*(\mathbf{y}) = \mathbf{b}^T \mathbf{y}, \\ & \text{subject to} \quad A^T \mathbf{y} \geq \mathbf{c} \\ & \text{and} \quad \mathbf{y} \geq 0. \end{array}$$

Weak Duality Theorem

Recall that any vector \mathbf{x} which satisfies the two constraints $A\mathbf{x} \leq \mathbf{b}$ and $\mathbf{x} \geq 0$ is called a *feasible solution*, regardless of what the corresponding objective value $\mathbf{c}^T \mathbf{x}$ might be.

Theorem

If $\mathbf{x} = \langle x_1 \dots x_n \rangle$ is any feasible solution for P and $\mathbf{y} = \langle y_1 \dots y_m \rangle$ is any feasible solution for P^* , then:

$$z(\mathbf{x}) = \sum_{j=1}^n c_j x_j \leq \sum_{i=1}^m b_i y_i = z^*(\mathbf{y})$$

Weak Duality Theorem

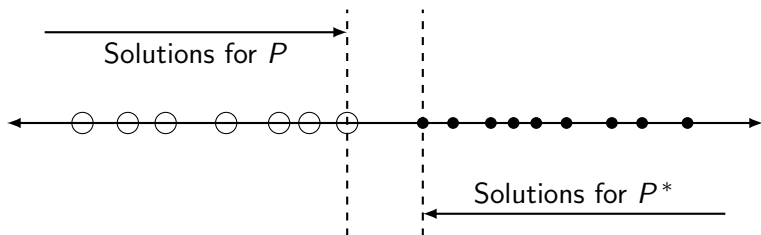
Proof

Since \mathbf{x} and \mathbf{y} are feasible solutions for P and P^* respectively, we can use the constraint inequalities, first from P^* and then from P to obtain

$$\begin{aligned} z(\mathbf{x}) &= \sum_{j=1}^n c_j x_j \leq \sum_{j=1}^n \left(\sum_{i=1}^m a_{ij} y_i \right) x_j \\ &= \sum_{i=1}^m \left(\sum_{j=1}^n a_{ij} x_j \right) y_i \leq \sum_{i=1}^m b_i y_i \\ &= z^*(\mathbf{y}). \end{aligned}$$

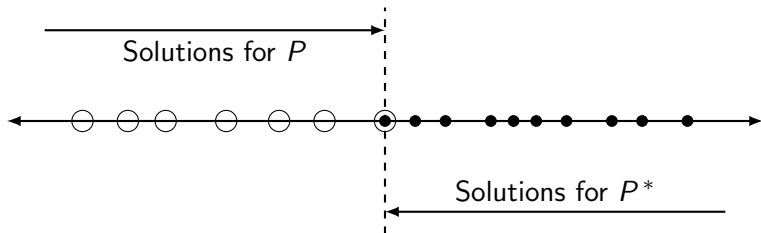
Weak Duality Theorem

Thus, the value of (the objective of P^* for) any feasible solution of P^* is an upper bound for the set of all values of (the objective of P for) all feasible solutions of P , and every feasible solution of P is a lower bound for the set of feasible solutions for P^* .



Weak Duality Theorem

If we find a feasible solution for P which is equal to a feasible solution to P^* , this common value must be the maximal feasible value of the objective of P and the minimal feasible value of the objective of P^* .



Weak Duality Theorem

- If we use a search procedure to find an optimal solution for P we know when to stop: when such a value is also a feasible solution for P^* .
- This is why the most commonly used LP solving method, the SIMPLEX method, produces an optimal solution for P : because it stops at a value of the primal objective which is also a value of the dual objective.
- See the supplemental notes for the details and an example of how the SIMPLEX algorithm runs.

Table of Contents

1. Example Problems
2. Linear Programming
3. Puzzle

Puzzle

There are five sisters in a house.

- Sharon is reading a book.
- Jennifer is playing chess.
- Catherine is cooking.
- Anna is doing laundry.

What is Helen, the fifth sister, doing?



That's All, Folks!!